Exercise walk-through

6.1
Data types, strings, exceptions and trace tables

International Baccalaureate · IB Diploma · Computer Science · HL

You must be able to

- choose a suitable **data type** for a value, and **cast** between types
- use string indexing, slicing and methods, and state what they return
- tell a **syntax error** from a **runtime error** and a **logic error**

- handle invalid input with `try` … `except`
- complete a **trace table** for code with a loop and selection

1 The method

Method — Key knowledge: types and strings

Type	Python	Example
integer	`int`	`42`
real	`float`	`3.75`
Boolean	`bool`	`True`
string	`str`	`"Ada"`

- `input()` always returns a string: `int(input())` casts it to an integer.
- `7 / 2` is `3.5`; `7 // 2` is `3`; `7 % 2` is `1`.

Exercises 2.1·2.2

2.1 State the most suitable data type for: (i) the number of seats in a cinema; (ii) the price of a ticket, 8.50; (iii) whether a seat is booked; (iv) a booking reference such as `"K7-204"`. [4]

2.2 Evaluate: (i) `17 // 5`; (ii) `17 % 5`; (iii) `17 / 5`. [3]

Exercises 2.1·2.2

2.1 State the most suitable data type for: (i) the number of seats in a cinema; (ii) the price of a ticket, 8.50; (iii) whether a seat is booked; (iv) a booking reference such as `"K7-204"`. [4]

2.1

- (i) **integer**
- (ii) **real** (float)
- (iii) **Boolean**
- (iv) **string**: it holds letters and a hyphen

2.2 Evaluate: (i) `17 // 5`; (ii) `17 % 5`; (iii) `17 / 5`. [3]

2.2

- (i) **3**
- (ii) **2**
- (iii) **3.4**

Exercises 2.3·2.4

2.3 `code = "IB2027CS"`. State: (i) `len(code)`; (ii) `code[2:6]`; (iii) `code.find("C")`; (iv) `code[-1]`. [4]

2.4 Classify each error as syntax, runtime or logic. [3]

(i) `print("Total:", total` with no closing bracket

(ii) an average calculated as `total / 10` when there are 12 items

(iii) `marks[5]` when `marks` has 5 items

Exercises 2.3·2.4

2.3 `code = "IB2027CS"`. State: (i) `len(code)`; (ii) `code[2:6]`; (iii) `code.find("C")`; (iv) `code[-1]`. [4]

2.3

- (i) **8**
- (ii) indexes 2 to 5: **`"2027"`**
- (iii) **6**
- (iv) the last character, **`"S"`**

2.4 Classify each error as syntax, runtime or logic. [3]

(i) `print("Total:", total` with no closing bracket

(ii) an average calculated as `total / 10` when there are 12 items

(iii) `marks[5]` when `marks` has 5 items

2.4

- (i) **syntax**
- (ii) **logic**: it runs but gives the wrong answer
- (iii) **runtime**: an index error, as valid indexes are 0 to 4

Exercises 3.1·3.2

3.1 What is printed by `print("3" + "4")`? [1]

- ☐ **A** `7`
- **B** `34`
- **C** `"3" + "4"`
- **D** an error message

3.2 `word = "python"`. Which expression gives `"tho"`? [1]

- ☐ **A** `word[2:4]`
- **B** `word[2:5]`
- **C** `word[3:5]`
- **D** `word[3:6]`

Exercises 3.1·3.2

3.1 What is printed by `print("3" + "4")`? [1]

- ☐ **A** `7`
- ☐ **B** `34`
- ☐ **C** `"3" + "4"`
- ☐ **D** an error message

3.1 B

- `+` joins two strings

3.2 `word = "python"`. Which expression gives `"tho"`? [1]

- ☐ **A** `word[2:4]`
- ☐ **B** `word[2:5]`
- ☐ **C** `word[3:5]`
- ☐ **D** `word[3:6]`

3.2 B

- indexes 2, 3, 4: `t`, `h`, `o`

Exercise 3.3

3.3 Which input makes `int(input())` raise a `ValueError`? [1]

- ☐ **A** `42`
- **B** `-7`
- **C** `007`
- **D** `4.5`

Exercise 3.3

3.3 Which input makes `int(input())` raise a `ValueError`? [1]

- ☐ **A** `42`
- **B** `-7`
- **C** `007`
- **D** `4.5`

3.3 D

- `int` cannot convert text with a decimal point

Setup for exercise 3.4

3.4 A program reads temperatures until the user enters `x`.

```
count = 0
total = 0
high = -100
t = input()
while t != "x":
    v = int(t)
    count = count + 1
    total = total + v
    if v > high:
        high = v
    t = input()
print(total / count, high)
```

Exercise 3.4 (a)

(a) Complete a trace table for the inputs `18`, `23`, `19`, `x`, with columns `t`, `count`, `total` and `high`, and state the output. [4]

Exercise 3.4 (a)

(a) Complete a trace table for the inputs `18`, `23`, `19`, `x`, with columns `t`, `count`, `total` and `high`, and state the output. [4]

(a)

`t`	`count`	`total`	`high`
`"18"`	0	0	-100
	1	18	18
`"23"`	2	41	23
`"19"`	3	60	23
`"x"`			

- `count` and `total` correct
- `high` correct
- loop ends at `x`
- output **20.0 23**

Exercise 3.4 (b)

(b) State the output if the first input is `x`, and explain why. [2]

Exercise 3.4 (b)

(b) State the output if the first input is `x`, and explain why. [2]

(b)

- the loop body never runs, so `count` is 0
- `total / count` divides by zero: a **`ZeroDivisionError`**, and nothing is printed

Exercise 3.4 (c)

(c) Explain why the program crashes if the user types `twenty`, and describe how `try … except` would prevent it. [3]

Exercise 3.4 (c)

(c) Explain why the program crashes if the user types `twenty`, and describe how `try … except` would prevent it. [3]

(c)

- `int("twenty")` raises a `ValueError`
- put the cast inside `try`
- the `except ValueError` block prints a message and asks again instead of stopping

Exercise 3.4 (d)

(d) Give **three** test inputs for the temperature, one normal, one boundary and one invalid, for a sensor that reads from -40 to 50. [3]

Exercise 3.4 (d)

(d) Give **three** test inputs for the temperature, one normal, one boundary and one invalid, for a sensor that reads from -40 to 50. [3]

(d)

- normal: e.g. 21
- boundary: -40 or 50
- invalid: e.g. 51, -41 or `"abc"`